1. You need a database system for a company with a massive physical warehouse operation. They want to keep track of their inventory, recording every shipment in and every shipment, with well-defined descriptions of each item and where they are stored. They also want to be able to analyze their operations to answer questions such as whether certain items should be stored closer together, or how often a particular item sells out. Which of the following would be the best choice for this? **1 / 1 point**

- (●) A non-transactional operational system designed for structured data, such as Apache Kudu
- ( ) A search system, such as Cloudera Search or Elasticsearch
- ( ) An analytic system (data warehouse) such as Apache Impala
- ( ) A non-transactional operational system designed for unstructured or semi-structured data, such as Apache HBase or MongoDB
- ( ) An ACID-compliant RDBMS for big data, such as Splice Machine or Apache Phoenix

✓ **Correct**

Correct. This is a good option to track the minute-to-minute operations and also allow the analytic questions to be explored.

2. You need a database system for a library of millions of large text documents, to help users find the documents that contain the information they need. Which of the following would be the best choice for this? **1 / 1 point**

- ( ) An ACID-compliant RDBMS for big data, such as Splice Machine or Apache Phoenix
- ( ) A non-transactional operational system designed for unstructured or semi-structured data, such as Apache HBase or MongoDB
- ( ) An analytic system (data warehouse) such as Apache Impala
- ( ) A non-transactional operational system designed for structured data, such as Apache Kudu
- (●) A search system, such as Cloudera Search or Elasticsearch

✓ **Correct**

Correct. The indexing and imprecise matching of a search system would provide a quick way to filter through the documents, searching for information that might not be well defined.

3. **1 / 1 point**

Which of the following are features of SQL on RDBMSs that are also kept for working with big data systems? Check all that apply.

- [x] **SELECT** statements

Correct

Correct. Queries using **SELECT** are the most the primary strength of SQL, and these are kept for working with big data.

- [ ] Support for many file formats
- [ ] Synchronized indexes
- [x] Seeing data as tables with column names

Correct

Correct. This is necessary to use SQL, so it is something we must retain when moving to big data.

- [ ] Unique values within columns

4. Which of the following is the reason why we lose many features of SQL when moving from traditional RDBMSs to big data systems? **1 / 1 point**

- (•) Many of the lost features require transactions, which are notoriously challenging for big data systems and not typically implemented
- ( ) Many of the lost features do not work well with the variety of data available in big data stores
- ( ) Many of the lost features are useful for relatively small amounts of data, but they become irrelevant for large volumes of data
- ( ) Many of the lost features are rarely used and implementation has been a low priority for big data systems

Correct

Correct. The loss of transactions accounts for most of the features lost in moving to SQL on big data systems.

5. **1 / 1 point**

Which of the following are features of SQL for working with traditional RDBMSs that we lose when moving to working with big data systems? Check all that apply.

- [ ] Complex data types
- [ ] **GRANT** and **REVOKE** statements
- [ ] Support for many file formats
- [x] Foreign key constraints

✓ **Correct**

Correct.

- [x] Database triggers and stored procedures

✓ **Correct**

Correct.

- [ ] Seeing data as tables with column names

6. Which of the following are features of SQL for working with big data systems that are not typically found in SQL for traditional RDBMSs? Check all that apply. **1 / 1 point**

- [ ] **UPDATE** and **DELETE** statements
- [ ] **GRANT** and **REVOKE** statements
- [ ] **CREATE** and **ALTER** statements
- [ ] Primary key constraints
- [x] Complex data types

✓ **Correct**

Correct. Complex data types make it easier to use denormalized tables, which is more common with big data than with traditional RDBMSs.

7. A company has a small on-premises cluster that they are rapidly outgrowing, and they are considering switching to cloud storage, or maintaining a hybrid solution. The following describes some factors going into their decision. Which are reasons that potentially support using a cloud cluster rather than an on-premises cluster? (Note that a hybrid option might still be best!) **0 / 1 point**

- [ ] The company's assets and budgets afford extra room for operating expenditures, but they are trying to keep capital expenditures to a minimum
- [x] The company hopes some upcoming new products will drastically increase their storage needs, though their processing needs probably will increase less dramatically

✓ **Correct**

Correct. This supports a separation of storage and processing, which you get with cloud-based solutions.

- [x] The company's analytics team processes queries nearly constantly, some ad-hoc during business hours and some bulk processes that run every day during off-hours; they do not experience significant periods of inactivity

! **This should not be selected**

Incorrect. You might want to review the "Where to Store Big Data" video.

- [ ] The company expects to continue to maintain and expand their data store for several years.

8. Which of the following accurately describes how the data dictionary in a traditional RDBMS being *tightly coupled* to the data is different from the table definitions in a big data system being *loosely coupled* to the files? Check all that apply. **1 / 1 point**

- [ ] The contents of the database are compressed for optimal storage space in a RDBMS, while the files in a big data system can be compressed if desired using the file format settings
- [x] The data dictionary governs what is stored as data in a RDBMS, while the files in a big data system are completely ungoverned

✓ **Correct**

Correct. The data dictionary is used to verify data when it's stored (schema on write), while the table definitions in a big data system don't touch data when it's being stored—only when it's being loaded in for a query (schema on read).

- [ ] Each table in a RDBMS has its own data dictionary, so they come in pairs, while a table definition in a big data system can be applied to different data files (a "one-to-many" coupling)

- [x] The contents of the data dictionary accurately describe every table in a RDBMS, while the table definitions in a big data system describe what is expected in some files, but even those files may not match exactly

Correct

Correct. The data in a RDBMS's tables must be only what is described by the data dictionary. The table definitions in a big data system provide a structure, but there is no guarantee that any of the files will follow that structure. In addition, some files may not be associated with any tables at all.